Exercícios

Exercício 1: Carrinho de compras e pagamento

Refatore o código a seguir utilizando os princípios :

- **Responsabilidade Única** *(SRP)*
- **Aberto/Fechado** (OCP)

```
class CarrinhoDeCompras {
    private List<String> itens = new ArrayList<>();

    public void adicionarItem(String item) {
        itens.add(item);
    }

    public void pagar(String metodo) {
        if (metodo.equals("cartao")) {
            System.out.println("Pagando com cartão de crédito...");
        } else if (metodo.equals("paypal")) {
            System.out.println("Pagando com PayPal...");
        } else {
            System.out.println("Método de pagamento não suportado.");
        }
    }
}
```

Exercício 2: Carrinho de compras e descontos

Refatore o código a seguir utilizando os princípios :

- **Responsabilidade Única** *(SRP)*
- **Aberto/Fechado** (OCP)

```
class CarrinhoDeCompras {
    private List<Double> precos = new ArrayList<>();

    public void adicionarItem(double preco) {
        precos.add(preco);
    }

    public double calcularTotal() {
        double total = 0;
        for (double preco : precos) {
            total += preco;
        }
        return total;
    }

    public double aplicarDesconto(String tipo) {
        if (tipo.equals("natal")) {
            return calcularTotal() * 0.9; // 10% de desconto
        } else if (tipo.equals("black_friday")) {
            return calcularTotal() * 0.8; // 20% de desconto
        }
        return calcularTotal();
    }
}
```

Exercício 3: Envio e cálculo de frete

Refatore o código a seguir utilizando os princípios :

- **Responsabilidade Única** *(SRP)*
- **Aberto/Fechado** (OCP)

```
class CarrinhoDeCompras {
    private List<String> itens = new ArrayList<>();
    private double pesoTotal;

    public void adicionarItem(String item, double peso) {
        itens.add(item);
        pesoTotal += peso;
    }

    public double calcularFrete() {
        if (pesoTotal < 5) {
            return 10.0;
        } else if (pesoTotal < 10) {
            return 20.0;
        } else {
            return 30.0;
        }
    }
}
```

Exercício 4: Controle de Estoque

Cenário: Uma loja deseja gerenciar seu estoque de diferentes tipos de produtos. Cada produto tem um identificador, nome e quantidade disponível. O sistema deve permitir a adição de novos produtos e atualização da quantidade.

Tarefas:

1. Criar uma estrutura para armazenar e gerenciar o estoque.
2. Permitir a adição de novos produtos ao estoque.
3. Criar um método para atualizar a quantidade de um produto existente.
4. Permitir o uso do sistema para diferentes tipos de produtos.

Exercício 5: Atraso de Entrega

Cenário: Uma empresa de logística deseja monitorar a entrega de pedidos. Cada entrega pode ter um status como "Em trânsito", "Entregue" ou "Atrasada". O sistema deve ser capaz de lidar com diferentes tipos de entregas, como envio normal ou expresso.

Tarefas:

1. Criar uma estrutura que represente uma entrega e seu status.
2. Criar um método que atualize o status de uma entrega.
3. O sistema deve permitir a adição de novos tipos de entrega sem modificar o código existente.
4. Criar um mecanismo para registrar quando uma entrega está atrasada.

Exercício 6: Estorno de Pagamento

Cenário: Um sistema de pagamentos precisa gerenciar estornos de diferentes formas de pagamento. O estorno pode ser realizado via cartão de crédito, transferência bancária ou outro método. Cada método de pagamento tem uma forma diferente de processar o estorno.

Tarefas:

1. Criar uma estrutura que represente um pagamento.
2. Criar um mecanismo que processe estornos de diferentes formas de pagamento.
3. Permitir que novos métodos de pagamento sejam adicionados sem modificar o código existente.
4. Implementar um sistema que registre os estornos realizados.